SNS COLLEGE OF TECHNOLOGY

Coimbatore-35.

An Autonomous Institution

COURSE NAME : 19CST201 AGILE SOFTWARE ENGINEERING

II YEAR/ III SEMESTER

UNIT – I INTRODUCTION TO SOFTWARE ENGINEERING

UNIT I INTRODUCTION TO SOFTWARE ENGINEERING

The Nature of Software -Software Engineering - Software engineering Practice – Process Models: Generic – Prescriptive – Specialized - United Process - Personal and Team Process Models - Process Technology-Understanding Requirements-Design concepts & model-Software quality concepts & Review metrics.

SOFTWARE DEFINITION

- Software is: (1) instructions (computer programs) that when executed provide desired features, function, and performance; (2) data structures that enable the programs to adequately manipulate information, and (3) descriptive information in both hard copy and virtual forms that describes the operation and use of the programs

1)INSTRUCTIONS/Programs:

- Functionality
- Performance

The INSTRUCTIONS must be developed according to the users satisfaction

2) DATA STRUCTURE:

- Essential Components
- Maintains Data
- Algorithms/ Program logic
- Design

3)DOCUMENTS:

- User Manual
- Design Methods

Software Types

- **System software**—a collection of programs written to service other programs.
- **Application software**—stand-alone programs that solve a specific business need.
- **Engineering/scientific software**—has been characterized by “number crunching” algorithms (numerical algorithms)

Software Types

- **Embedded software**—resides within a product or system and is used to implement and control features and functions for the end user and for the system itself.

- **Product-line software**—designed to provide a specific capability for use by many different customers.

Software Types

- **Web applications** —called “WebApps,” this network-centric software category spans a wide array of applications.
- **Artificial intelligence software** —makes use of non numerical algorithms to solve complex problems.

Software Engineering

Definition

*Software Engineering is the establishment and use of the **engineering principles** in order to obtain **economical software** that is reliable and work efficiently on real machines*

Software Engineering Activities

- Requirements gathering and Analysis
- Planning
- Design
- Development
- Testing
- Maintaining

Requirements gathering and Analysis

- Simple listing
- Surveys
- Interviews
- Focus
- Observation
- Use case Analysis

Types:

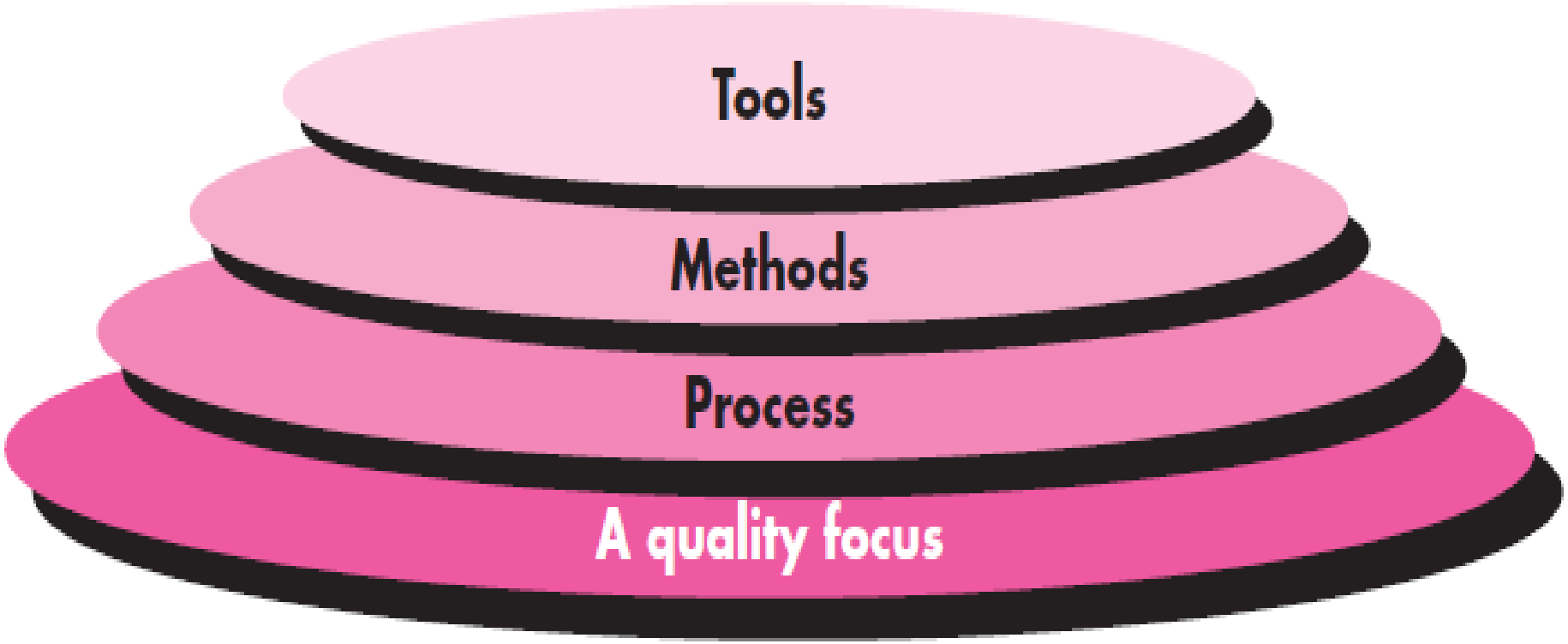
1)Functional Requirements –Something that the system must do;

Eg:Business rules of functions like add a customer ,print invoice etc

2)Non Functional Requirements –quality characteristics or attributes of the system

Eg: providing user access more than customers expectation

Software Engineering Layers



Software Engineering Practice

The Essence of Software Engineering Practice :

- Understand the problem (communication and analysis).
- Plan a solution (modeling and software design).
- Carry out the plan (code generation).
- Examine the result for accuracy (testing and quality assurance).

Software Process

- A software process is represented as **a set of work phase** that is applied to design and build a software product
- There is **no ideal software process** and many organization have developed their own approach to software development
- A process is a collection of activities, actions, and tasks that are performed when some work product is to be created.
- The intent is always to deliver software in a timely manner and with sufficient quality to satisfy the customers

Fundamental Activities of Software Process

There are some fundamental activities that are common to all software process :

- Software Specification
- Software design and implementation
- Software validation
- Software evolution

Software Process Framework

- A process framework establishes the foundation for a complete software engineering process by identifying a small number of framework activities that are applicable to all software projects, regardless of their size or complexity.

Software Process Model

- It’s a development strategy designed to solve an actual problem in an industry settings
  - Generic process model
  - Prescriptive process model
  - Specialized process models
  - The unified process
  - Personal and team process models

Generic Process Framework

A generic process framework for software engineering encompasses five activities:

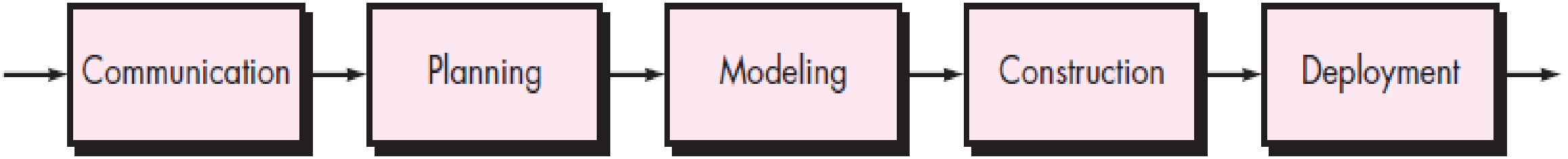
- **Communication**
- **Planning -** software project plan—defines the software engineering work by describing the technical tasks to be conducted, the risks that are likely, the resources that will be required, the work products to be produced, and a work schedule.
- **Modeling -** creating models to better understand software requirements
- **Construction -** combines code generation and the testing
- **Deployment**

Process Flow

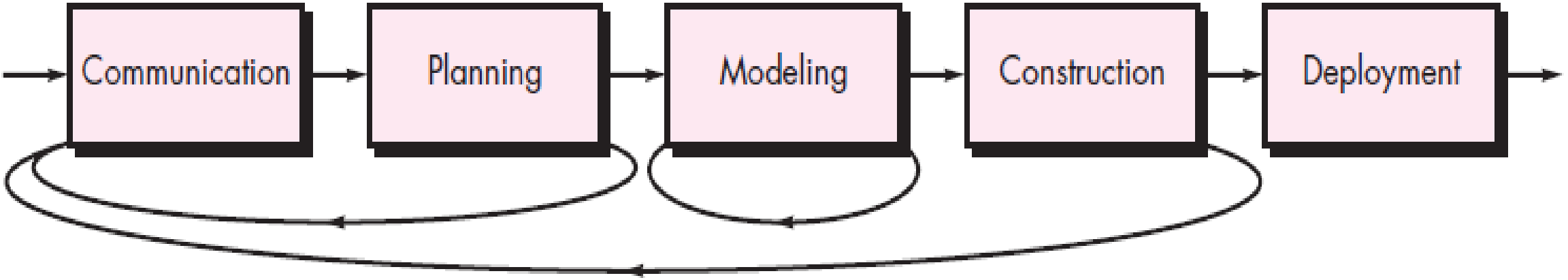
There are four types of process flow they are:

1. **Linear process flow -** executes each of the five framework activities in sequence
2. **Iterative process flow -** repeats one or more of the activities before proceeding to the next
3. **Evolutionary process flow -** executes the activities in a “circular” manner
4. **Parallel process flow -** executes one or more activities in parallel with other activities

Process Flow

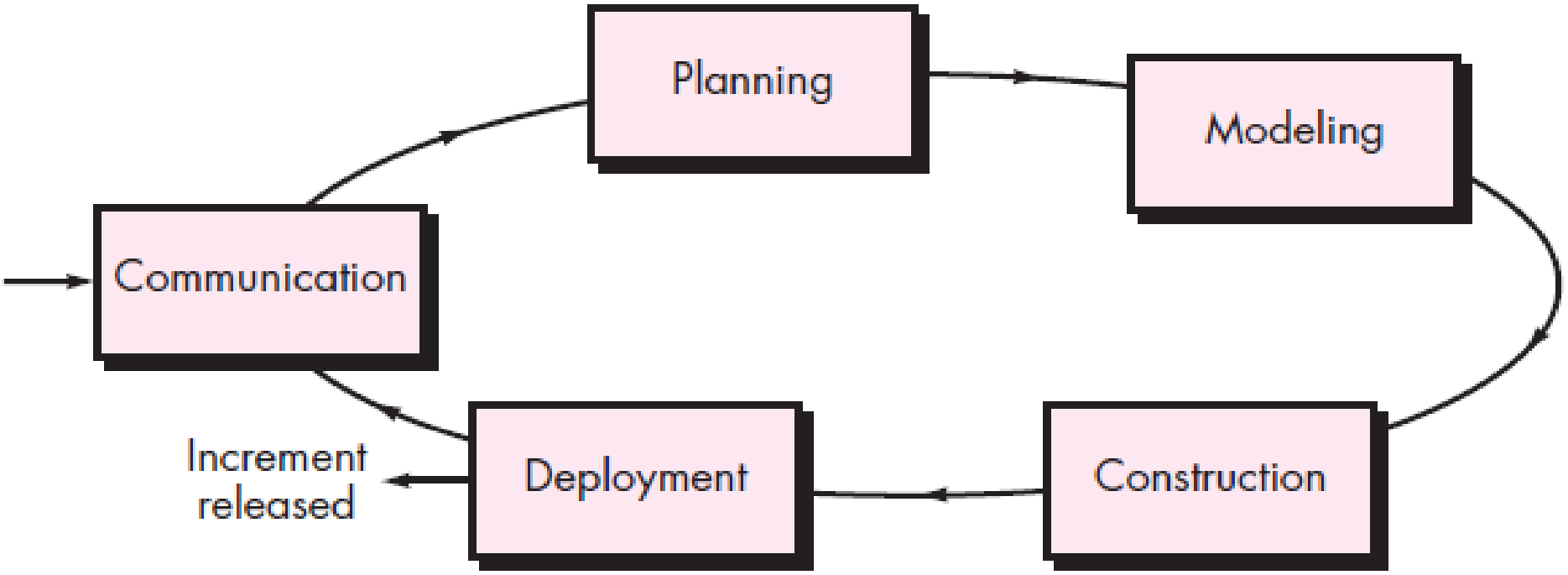


(a) Linear process flow



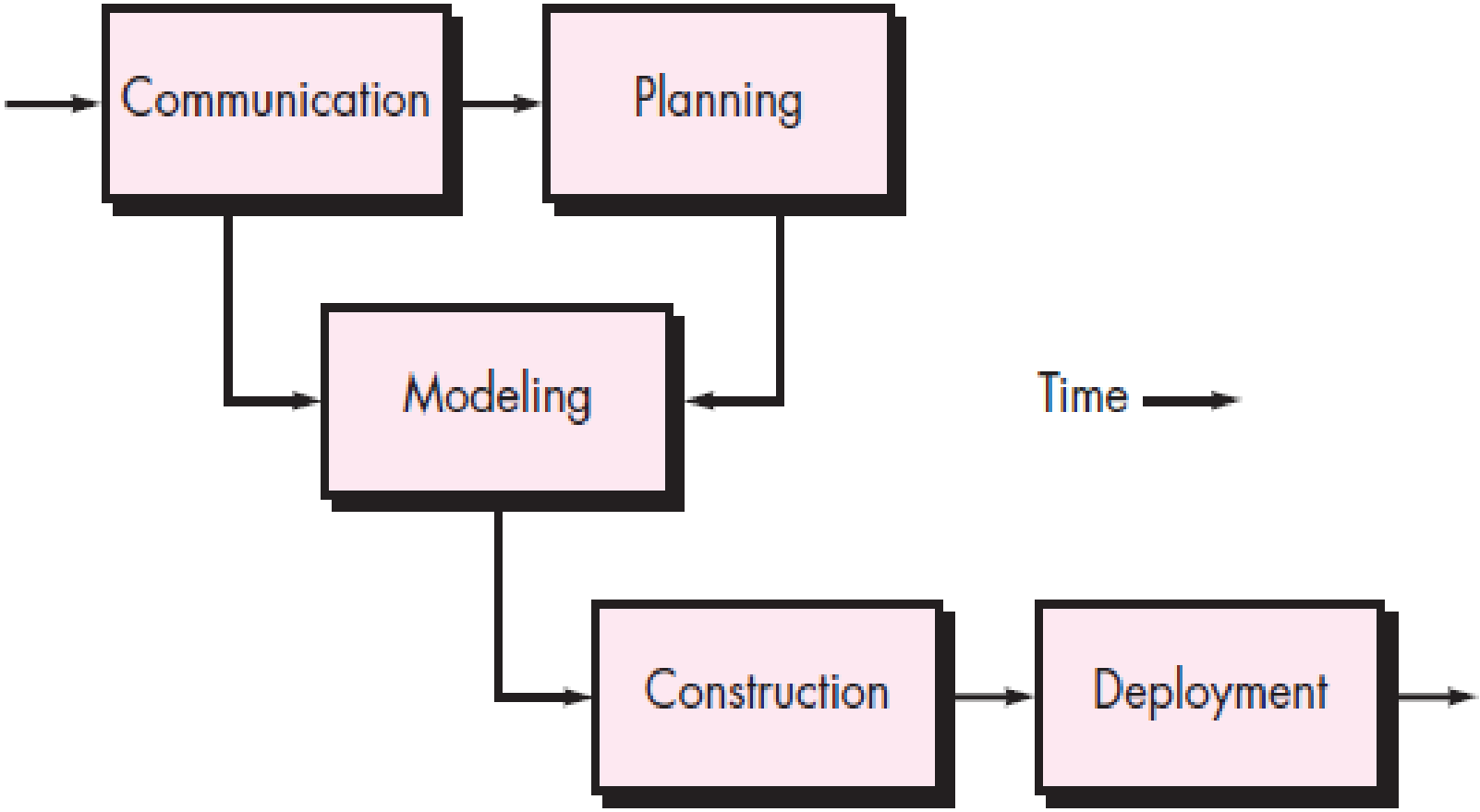
(b) Iterative process flow

Process Flow



(c) Evolutionary process flow

Process Flow



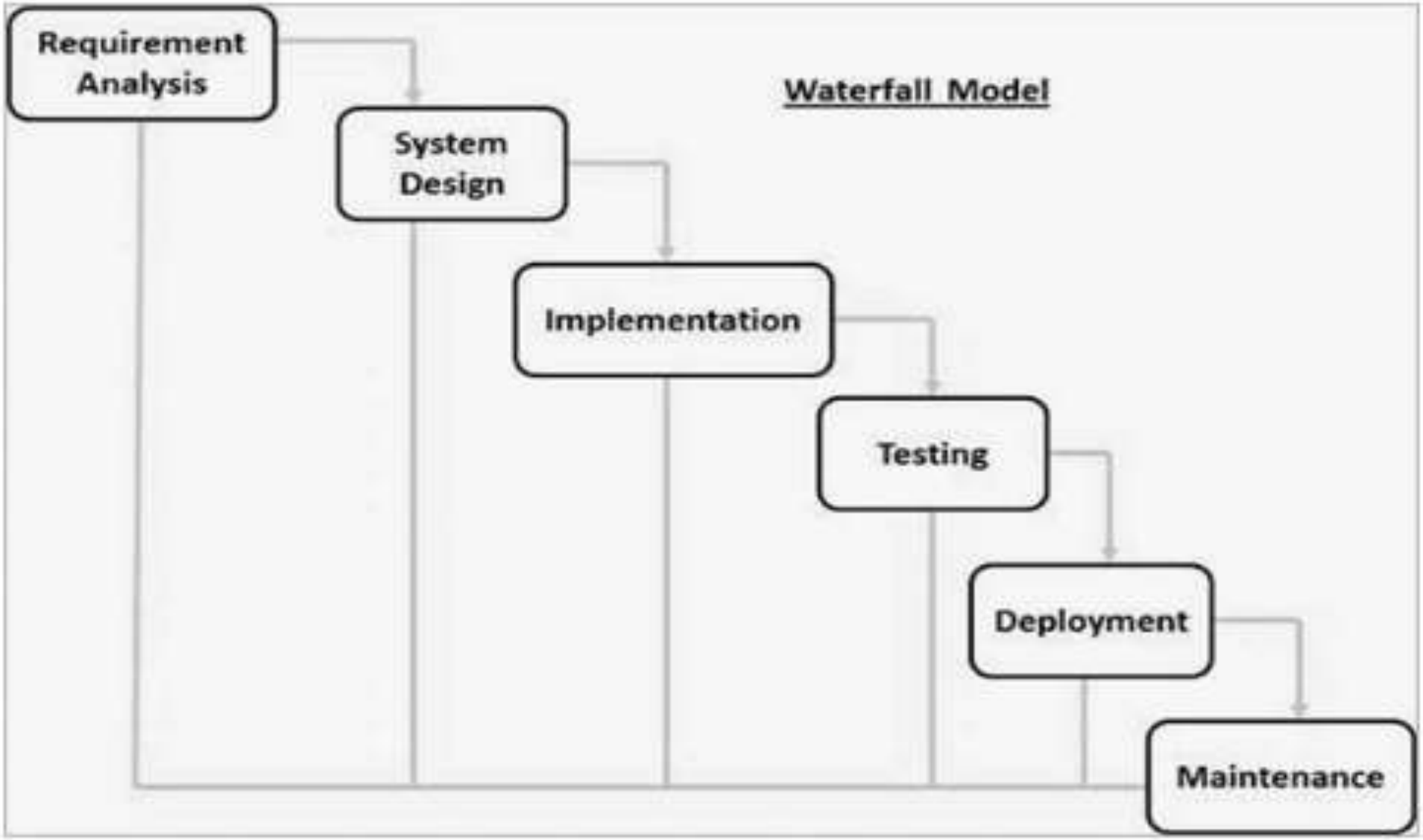
(d) Parallel process flow

Prescriptive Process Models

There are three types of prescriptive process models. They are:

1. The Waterfall Model
2. Incremental Process model
3. RAD model

The Waterfall Model



The Waterfall Model

The sequential phases in Waterfall model are −

- Requirement Gathering and analysis
- System Design − helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- Implementation − the system is first developed in small programs called units, which are integrated in the next phase.
- Integration and Testing − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- Deployment
- Maintenance − There are some issues which come up in the client environment to fix those issues, patches are released. Also to enhance the product some better versions are released.

The Waterfall Model

- The waterfall model is also called as **'Linear sequential model' or 'Classic life cycle model'.**
- In this model, each phase is fully completed before the beginning of the next phase.
- This model is used for the small projects.
- In this model, feedback is taken after each phase to ensure that the project is on the right path.
- Testing part starts only after the development is complete.

The Waterfall Model

Advantages :

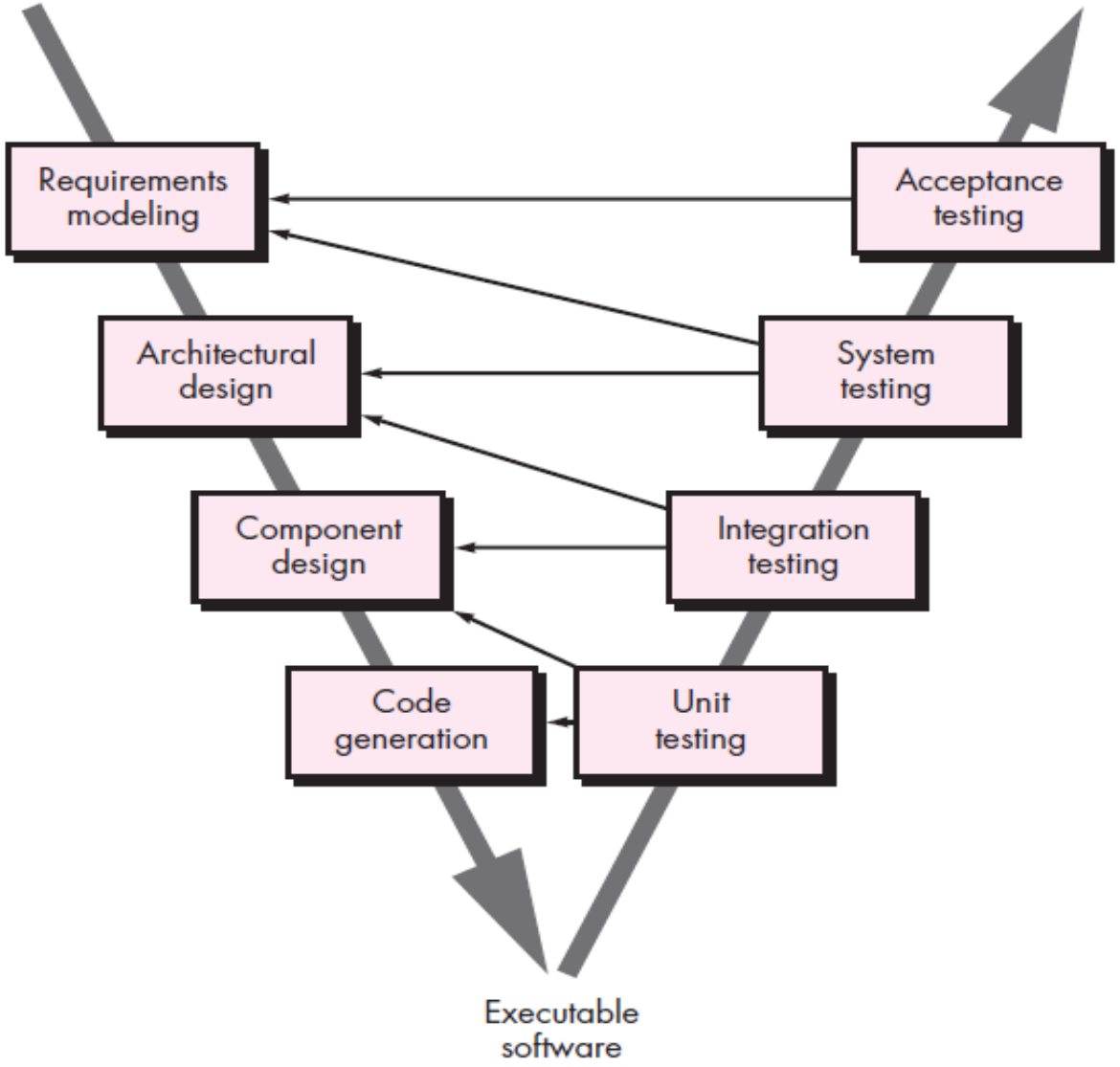
1. The waterfall model is simple and easy to understand, implement, and use.
2. All the requirements are known at the beginning of the project, hence it is easy to manage.
3. They should perform quality assurance test before completing each stage
4. Elaborate documentation is done at every phase of the software development cycle
5. Project is completely dependent on project team with minimum client intervention

The Waterfall Model

Disadvantages :

1. Error can be fixed only after the testing period
2. It is not suitable for a complex project
3. Documentation occupies a lot of time
4. Client valuable feedback cannot be included with ongoing development phase

V-Model


Requirements modeling
Architectural design
Component design
Code generation
Unit testing
Integration testing
System testing
Acceptance testing
Executable software

Incremental Process model

- The incremental model combines the elements of waterfall model and they are applied in an iterative fashion.

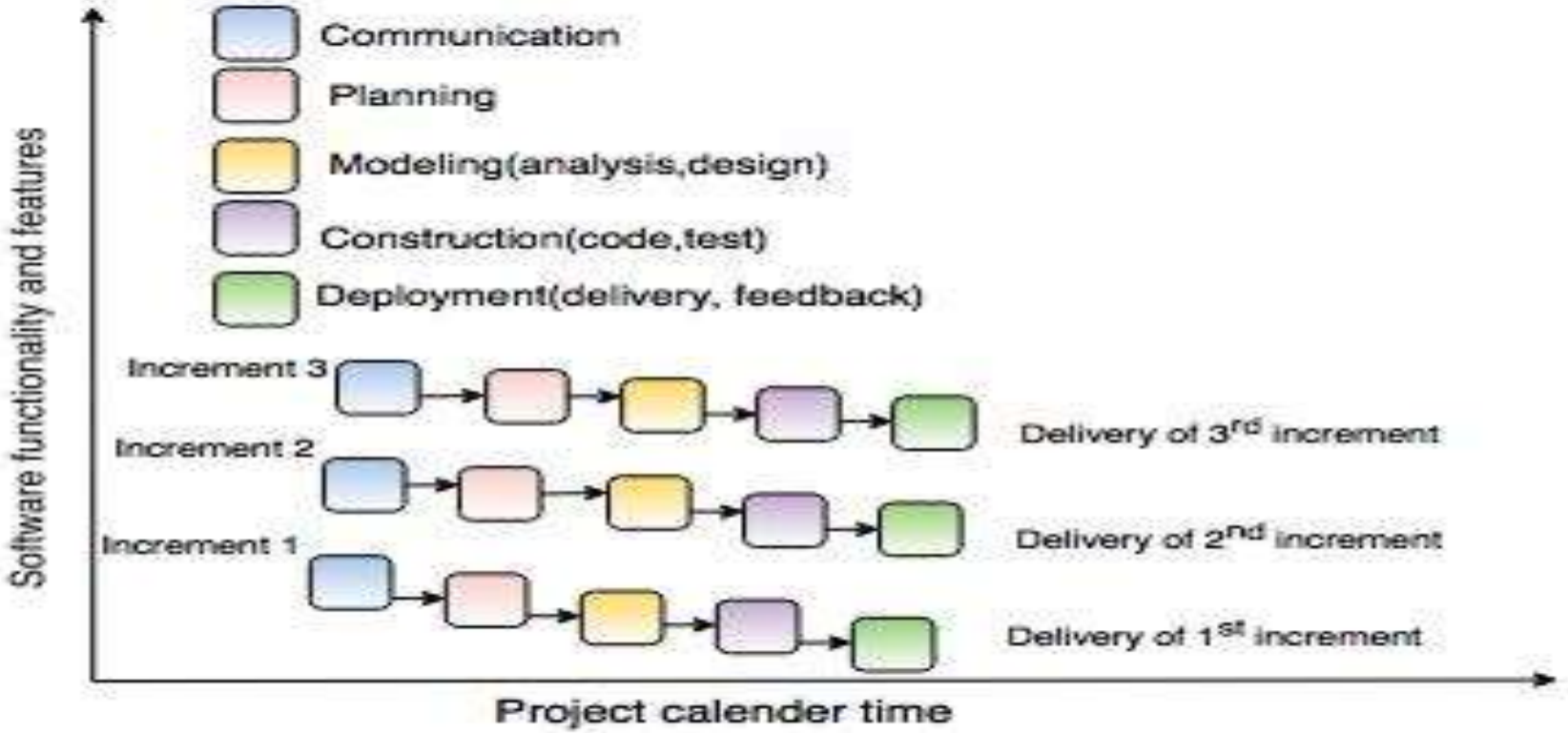


Fig. - Incremental Process Model

Incremental Process model

- The first increment in this model is generally a core product
- Each increment builds the product and submits it to the customer for any suggested modifications.
- The next increment implements on the customer's suggestions and add additional requirements in the previous increment.
- This process is repeated until the product is finished.

Incremental Process model

Advantages :

- This model is flexible because the cost of development is low and initial product delivery is faster
- It is easier to test and debug during the smaller iteration.
- The working software generates quickly and early during the software life cycle.
- The customers can respond to its functionalities after every increment.

Incremental Process model

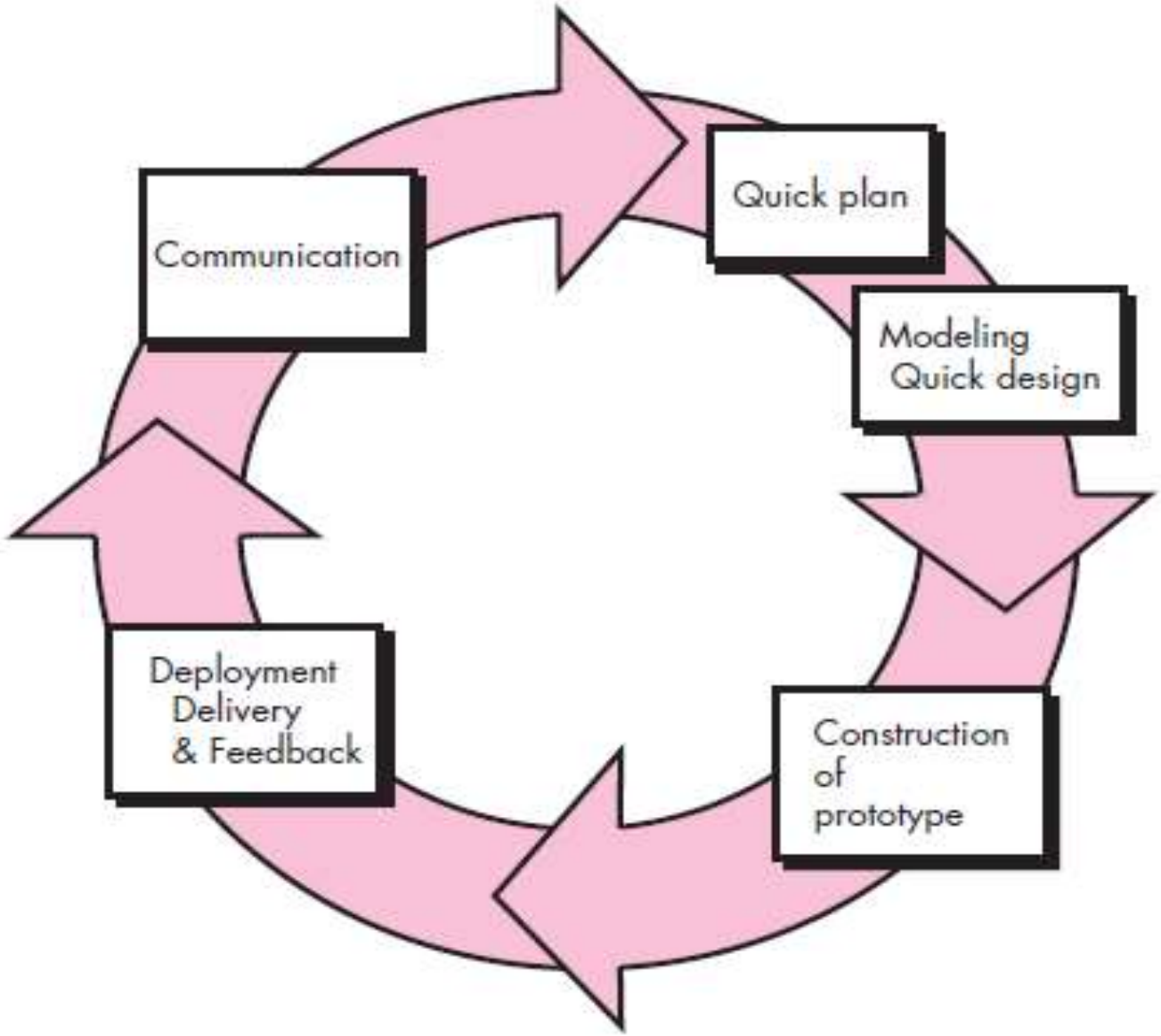
Disadvantages :

- Need clear planning and design
- The planning of design is required before the whole system is broken into small increments.
- Total cost is higher than water fall model

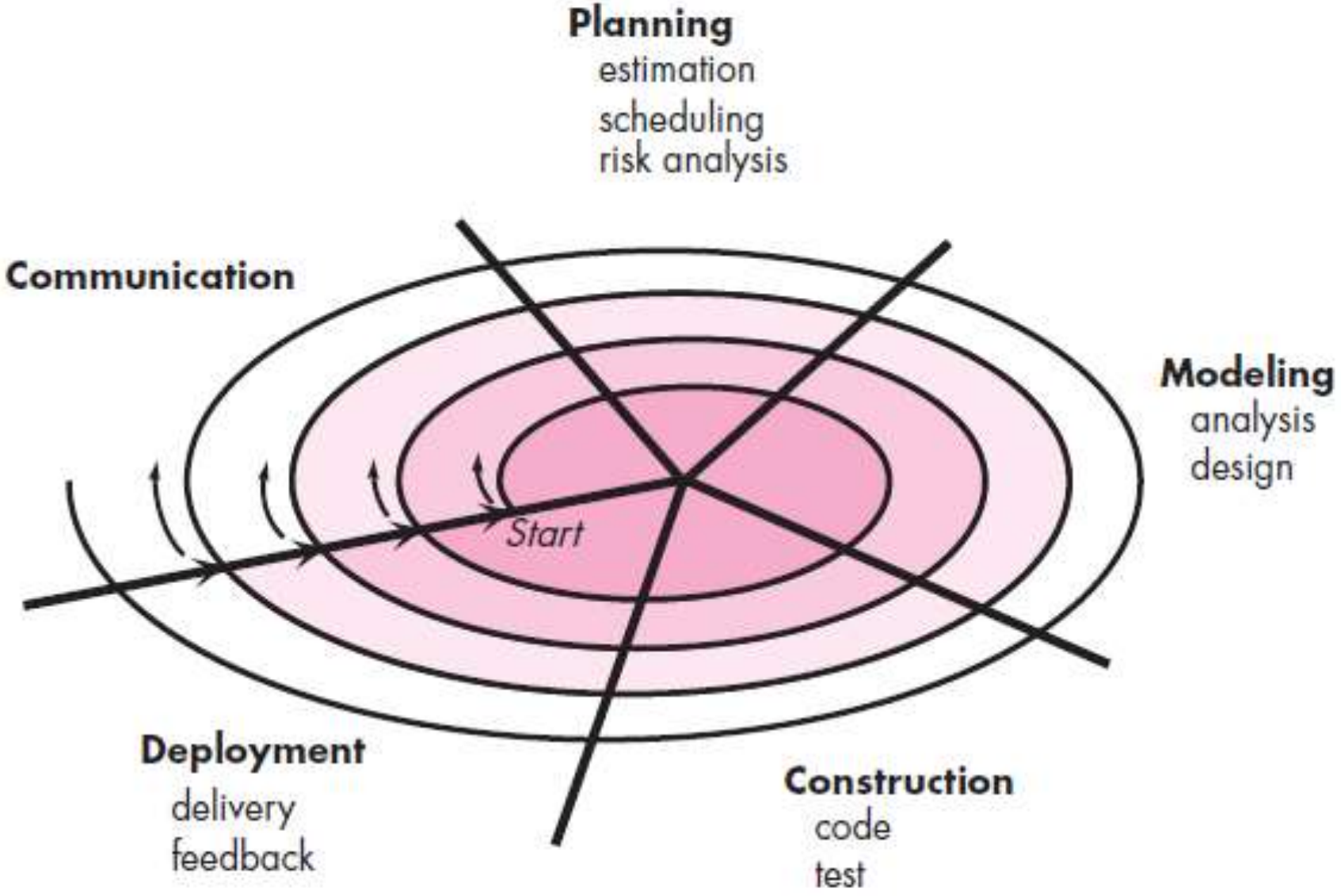
Evolutionary Process Models

- Evolutionary models are iterative. They are characterized in a manner that enables you to develop increasingly more complete versions of the software.
- Evolutionary process models :
    1. **Prototyping**
    2. **The Spiral Model**

Prototyping



The Spiral Model



The Spiral Model

- Proposed by Barry Boehm [Boe88].
- It is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the waterfall model.
- risk-driven process model
- cyclic approach
- anchor point milestones

RAD model

Rapid Application Development

- Using the RAD model, software product is developed in a short period of time.
- The initial activity starts with the communication between customer and developer.
- Planning depends upon the initial requirements and then the requirements are divided into groups model
- It is a **high speed** adaptation of the **linear sequential model** in which rapid development is achieved by using **component based construction**

Core Elements of RAD

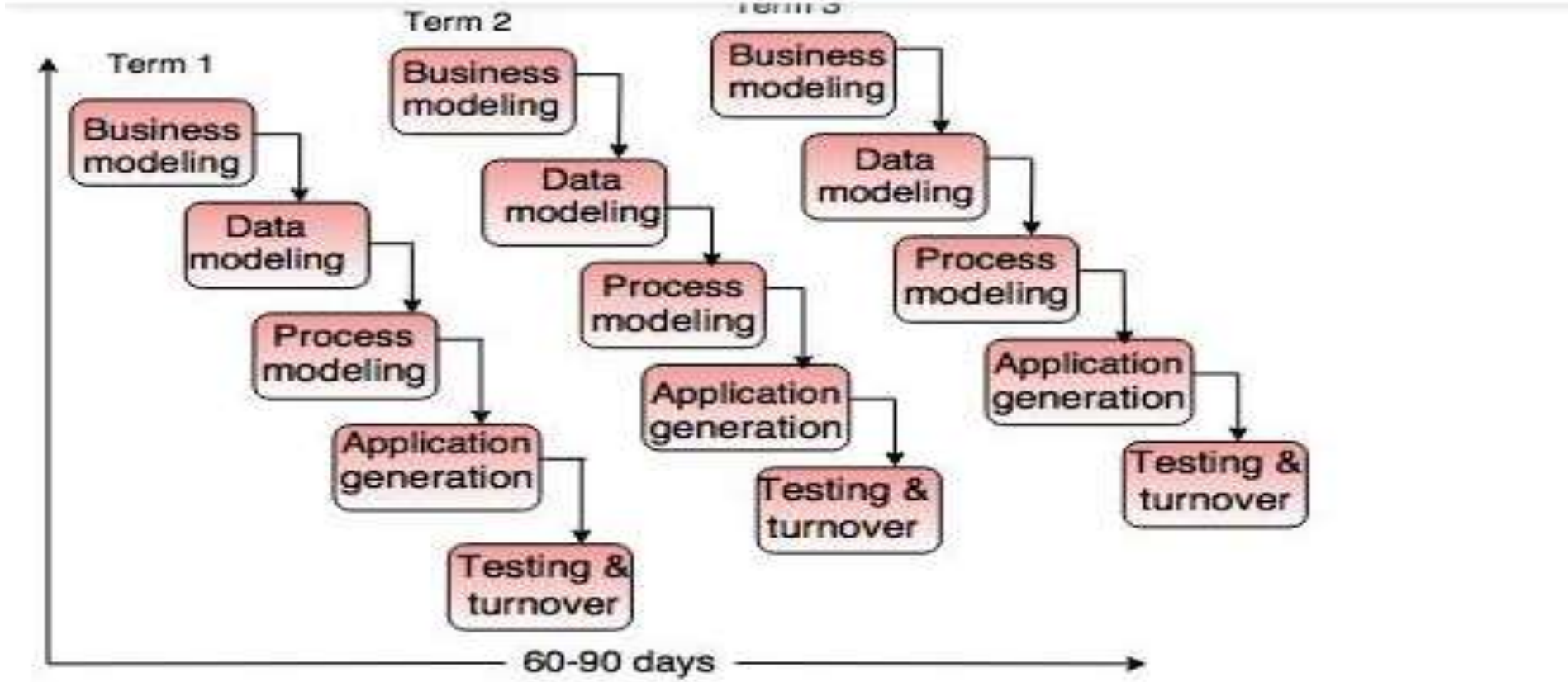


Fig. - RAD Model

Software Process Model

- It’s a development strategy designed to solve an actual problem in an industry settings
  - Generic process framework
  - Specialized process models
  - The unified process
  - Personal and team process models

SPECIALIZED PROCESS MODELS

- This model take on many of the characteristics of one or more of the traditional models
- These models tend to be applied when a specialized or narrowly defined software engineering approach is chosen.
  - **Component Based Development** (Promotes reusable components)
  - **The Formal Methods Model** (Mathematical formal methods are backbone here)
  - **Aspect Oriented Software Development (AOSD**)(use crosscutting technology)

Component Based Development

- Component Based Software Engineering (CBSE) is a process that focuses on the design and development of computer-based systems with the **use of reusable software components**
- Develop software using already available components
- In this kind of development there is no concept of building any software from scratch

Definition and characteristics of components

- Software component is a software package ,a web service, a web resource that encapsulates a set of related functions/data
- Developed components must be portable
- Replaceable/ Reusable

CBSE Framework Activities

1. Component Qualification:

- Ensures that the system architecture define the requirements of the components for becoming a reusable component.
- It means “the services that are given, and the means by which customers or consumers access these services ” are defined as a part of the component interface.

2. Component Adaptation:

- This activity ensures that the architecture defines the design conditions for all component and identifying their modes of connection.

CBSE Framework Activities

3. Component Composition:

- This activity ensures that the Architectural style of the system integrates the software components and form a working system.

4. Component Update:

- This activity ensures the updation of reusable components.

Characteristics of CBSE

- Reusability
- Replaceable
- Not context specific
- Extensible

The Formal Methods Model (Proof, Calculation , precision, Understanding)

- What is a FORMAL METHOD MODEL?
  - The Formal Methods Model is an approach to Software Engineering that applies **mathematical methods or techniques** to the process of developing complex software systems. The approach uses a formal specification language to define each characteristic of the system. .

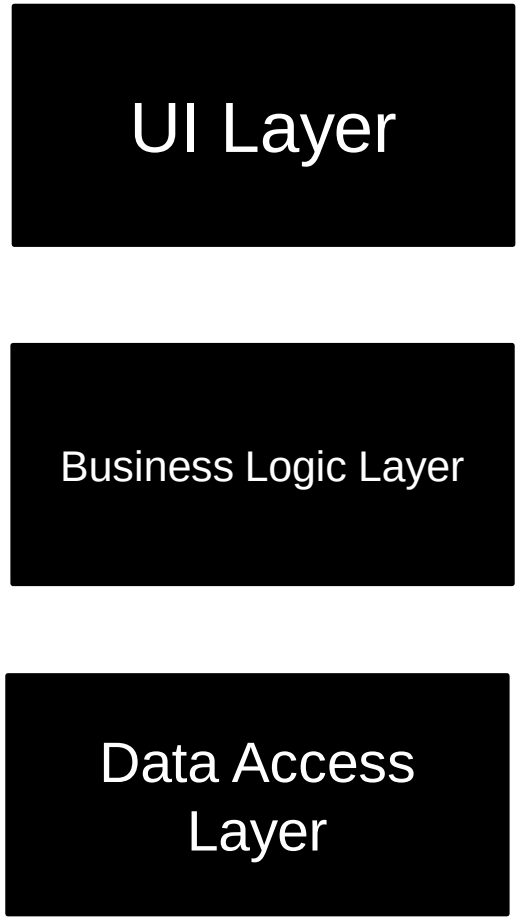
Formal methods can be useful in :

1. Articulating, and representing requirements .
2. Specifying software : developing a precise statement of what the software is to do .
3. Software design : Data refinement involves state machine specification, abstraction functions, and simulation proofs .
4. Coding verification
5. Enhancing early error detection .
6. Developing safe, reliable, secure software - intensive systems .
7. The overall effect of the use of formal techniques on time, cost ,and quality.

Formal Specification Methods

- Formal Specifications
- Formal Proofs
- Model Checking
- Abstraction

Aspect Oriented Software Development



- Security
- Profile
- Logging
- Transaction Management

Aspect Oriented Software Development

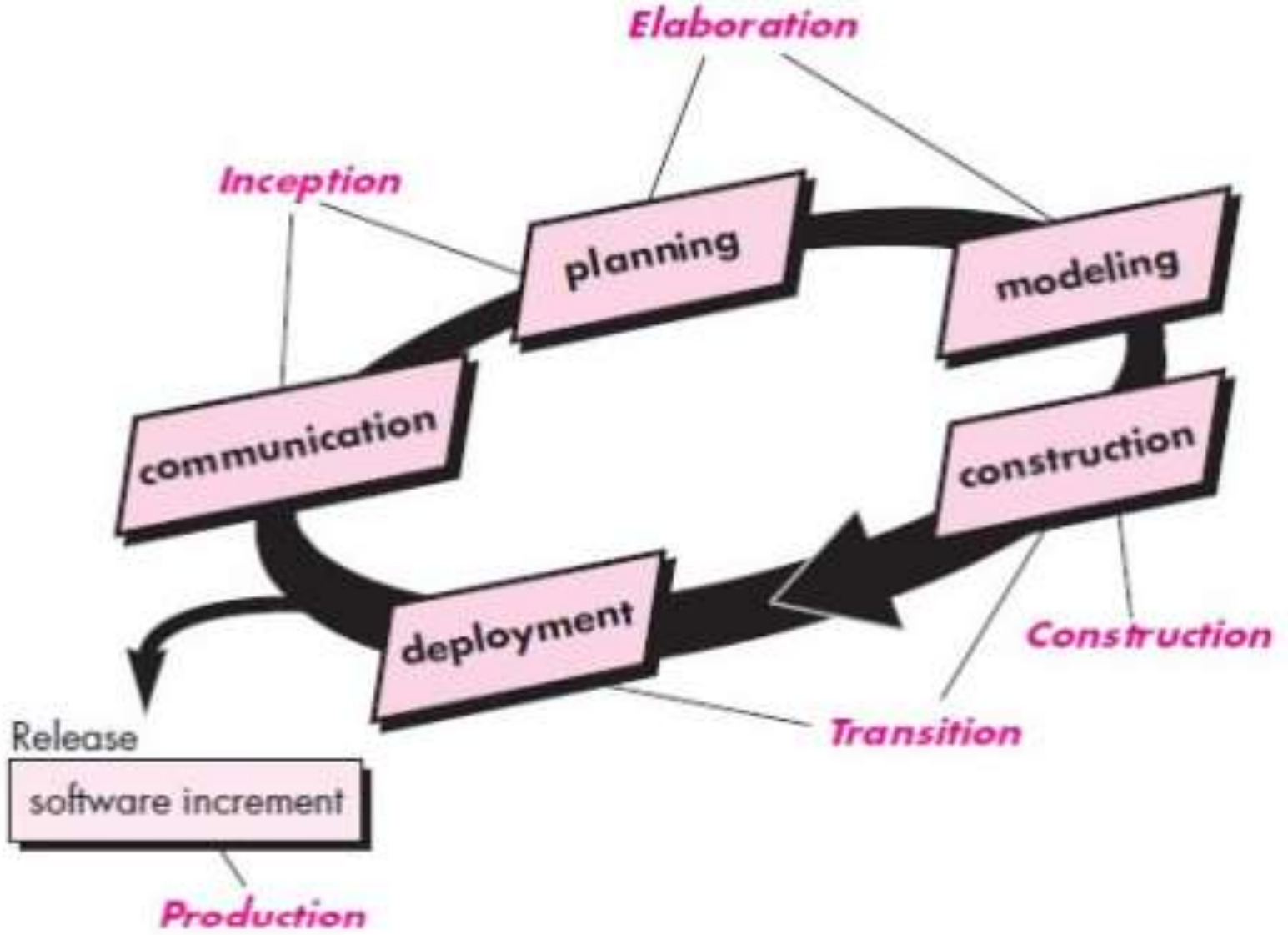
Advantages :

- Cross cutting Concern
- Reuse
- Quick Development
- Enabled /Disabled

Software Process Model

- It’s a development strategy designed to solve an actual problem in an industry settings
  - Generic process framework
  - Specialized process models
  - The unified process
  - Personal and team process models

Unified Process Model



Unified Process Model

- It implements many of the best principles of agile software development.

Phases of the Unified Process:

This process divides the development process into five phases:

1. Inception
2. Elaboration
3. Conception
4. Transition
5. Production

PERSONAL SOFTWARE PROCESS (PSP)

The aim of PSP is to give software engineers with the regulated methods for the betterment of personal software development processes.

The PSP helps software engineers to:

- Improve their approximating and planning skills.
- Make promises that can be fulfilled.
- Manage the standards of their projects.
- Reduce the number of faults and imperfections in their work.

PERSONAL SOFTWARE PROCESS (PSP)

PSP Framework Activities :

- Planning
- High Level Design
- High Level Design Review
- Development
- Postpartum

TEAM SOFTWARE PROCESS (TSP)

The Team Software Process (TSP), along with the Personal Software Process, helps the high- performance engineer to

- **ensure quality software products**
- **create secure software products**
- **improve process management in an organization**

TEAM SOFTWARE PROCESS (TSP)

TSP Framework Activities :

- Launch High Level Design
- Implementation
- Integration
- Test
- Postpartum


Thank You!